UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/626,420	07/24/2003	Sheueling Chang Shantz	6000-32301	9856

58467 7590 07/21/2010
MHKKG/Oracle (Sun)
P.O. BOX 398
AUSTIN, TX 78767

EXAMINER	
JOHNSON, CARLTON	

ART UNIT	PAPER NUMBER
2436	

NOTIFICATION DATE	DELIVERY MODE
07/21/2010	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

patent_docketing@intprop.com
ptomhkkg@gmail.com

Office Action Summary	Application No.	Applicant(s)
	10/626,420	SHANTZ ET AL.
	Examiner	Art Unit
	CARLTON V. JOHNSON	2436

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 04 May 2010.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-65 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-65 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☐ Notice of References Cited (PTO-892)

2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)

3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.

4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.

5) ☐ Notice of Informal Patent Application

6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to application amendments filed on 5-4-2010.

2. Claims **1 - 65** are pending. Claims **1, 18, 43, 50, 57, 61, 64, 65** have been amended. Claims **1, 18, 43, 50, 57, 61, 64, 65** are independent. This application was filed on 7-24-2003.

Response to Arguments

3. Applicant's arguments have been fully considered but they were not persuasive.

3.1 Applicant argues that the referenced prior art does not disclose, *add-chaining operation and multiply-chaining operation*.

The add-chaining operation and multiply-chaining operation discloses a combination (chaining) of accumulate and multiplication operations into a single arithmetic instruction. The Specification in paragraph [1076] states that the instruction performs multiply-accumulate-chaining operation, which combines (appears to be equivalent to prior art chaining) add-chaining and multiply-chaining in one operation in order to avoid the multiplier latency from the processing of operand(s) information between arithmetic operations. This statement appears to suggest the combination or chaining of two or more arithmetic operations with an implicit input or transfer of operand(s) between arithmetic instructions. Contrary to Applicant's assertions, the referenced prior art appears to disclose an equivalent operation as the claimed invention. The Examiner is not mischaracterizing the teachings of Applicant's Specification. The concept of

combining arithmetic operations is disclosed in the specification and the referenced prior art. The concept of the implicit transfer of operands between instructions is disclosed in the specification and the referenced prior art. The concept of utilizing arithmetic operations in the generation of an encryption key for cryptographic calculations are disclosed in the specification and the referenced prior art.

The concept of the claimed invention is to combine a set of arithmetic operations (accumulate and multiplication arithmetic operations) into a single arithmetic instruction or invocation. An arithmetic operation generates a resultant operand which is transferred between two (successive) arithmetic operations. The transfer of this generated operand is the feedback mechanism indicated by the claimed invention. The resultant or feedback operand is implicitly (without an explicit operation) transferred to the next arithmetic operation or instruction in the sequence. Applicant's invention discloses that the resultant operand can be a partial result (high-order or low-order bits from a resultant word.

Huppenthal discloses an architecture for chaining a number of arithmetic operations (such as multiplication, accumulation, and etc) to form a single arithmetic instruction. The single arithmetic instruction is initiated and the sequence of chained operations is performed with operand transfer controlled by the computing system architecture via the usage of a chain port mechanism. The chain port mechanism supplies operands to each successive arithmetic operation in the sequence. The chain port mechanism supplies operands without any support from the processor. Hinds discloses the

generation of a partial result (high-order or low-order bits). Huppenthal and Hinds disclose chaining a set of two or more arithmetic operations within a single arithmetic instruction and the generation of a partial result as the operand that is implicitly transferred between the arithmetic instructions.

3.2 Applicant argues that the referenced prior art does not disclose, *add-chaining and multiply-chaining as part of an instruction set architecture.*

Huppenthal and Hinds disclose the concept of chaining of arithmetic operations. This chaining concept is part of the architecture of the computing system disclosed within the Huppenthal prior art claimed invention. The chain operation is a legitimate operation within the instruction set of the Huppenthal prior art.

3.3 Applicant argues that the referenced prior art does not disclose, *obviousness rejection.*

A 103 rejection based on multiple references is a legitimate technique according to the MPEP. The current application is rejected based on the Huppenthal, Hinds and Chen prior art references. The set of references are in a same field of endeavor as the claimed invention, computing system instruction set processing. The 103 rejection allows portions of a claimed invention to come from different prior art references. The prior art references are within the same field of high performance computations which can be used for cryptographic operations such as encryption key generation.

3.4 Applicant argues that the referenced prior art does not disclose, *Independent Claims 43, 57, 64.*

Independent claims 43, 57, 64 have similar limitations as independent claim 1. Reponses to arguments for independent claim 1 respond to arguments against independent claims 43, 57, 64.

3.5 Applicant argues that the referenced prior art does not disclose, *Independent Claims 50, 61,65.*

Independent claims 18, 50, 61, 65 have similar limitations as independent claim 1. Reponses to arguments for independent claim 1 respond to arguments against independent claims 50, 61, 65.

Claim Rejections - 35 USC § 101

4. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

5. Claims **57 - 63** are rejected under 35 USC 101 since the claims are directed to non-statutory subject matter. Claims **57, 61** recite computer-readable storage medium which appear to cover both transitory and non-transitory embodiments. The broadest reasonable interpretation of a claim drawn to a computer readable media (also called machine readable medium and other such variations) typically covers forms of non-transitory tangible media <u>and</u> transitory propagating *signals per se* in view of the ordinary and customary meaning of computer readable medium.

The Examiner suggests that the Applicant add the limitation "non-transitory" to the computer readable storage medium as recited in the claim(s) in order to properly render the claim(s) in statutory form in view of their broadest reasonable interpretation in light of the originally filed specification. The Examiner also suggests that the specification may be amended to add the term "non-transitory computer-readable storage medium" to avoid a potential objection to the specification for a lack of antecedent basis of the claimed terminology.

Claim Rejections - 35 USC § 103

6. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

7. Claims **1, 2, 4, 5, 13 - 16, 18, 19, 21 - 23, 30, 33 - 44, 49 - 51, 56 - 65** are rejected under 35 U.S.C. 103 (a) as being unpatentable over **Huppenthal et al.** (US Patent No. **6,339,819**) in view of **Hinds et al.** (US Patent No. **6,542,916**) and further in view of **Chen et al.** (US Patent No. **6,763,365**).

With Regards to Claim 1, Huppenthal discloses a method implemented in a device, the method comprising: executing a single arithmetic instruction of a processor instruction set, and storing at least a portion of the generated result; and using the

stored at least a portion of the generated result in a subsequent computation in the cryptography application, and wherein the single arithmetic instruction does not include an explicit source operand for specifying the partial result. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port),

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses multiplying a first number by a second number; and adding implicitly a partial result from an executed arithmetic instruction to generate a result that represents the first number multiplied by the second number summed with the partial result, wherein the partial result comprises a high order portion of a result of the previously executed single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

It would have been obvious to one of ordinary skill in the art to modify Huppenthal for a multiplying and summing sequence as taught by Hinds. One of ordinary skill in the art would have been motivated to employ the teachings of Hinds to specifically increase

the speed of multiply-accumulate operations and achieve faster computations. (Hinds col. 1, lines 27-29)

Huppenthal-Hinds does not specifically disclose supporting a cryptographic application. However, Chen discloses supporting a cryptography application. (see Chen col. 6, lines 23-25: arithmetic operations to support acceleration of cryptographic functions),

It would have been obvious to one of ordinary skill in the art to modify Huppenthal-Hinds for supporting a cryptographic application as taught by Chen. One of ordinary skill in the art would have been motivated to employ the teachings of Chen to greatly improve the performance of cryptographic circuits. (Chen col. 5, lines 40-42)

With Regards to Claim 2, Huppenthal discloses the method as recited in claim 1 further comprising performing the adding of the partial result as part of addition operations performed for the multiplying of the first and second number. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 4, Huppenthal discloses the method as recited in claim 1, wherein said adding the partial result comprises adding the partial result to a multiplication result of the first and second numbers. (Hines col. 3, lines 5-17: applying

a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 5, Huppenthal discloses the method as recited in claim 1, wherein said storing at least a portion of the generated result comprises storing a high order portion of the generated result as a next partial result for use with execution of a subsequent single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 13, Huppenthal discloses the method as recited in claim 1, wherein the single arithmetic instruction is a single multiply-accumulate instruction; wherein the first and second numbers are specified in the single multiply-accumulate instruction as first and second source registers and a low order portion of the result is stored in a destination location specified in the single multiply-accumulate instruction.

(Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 14, Huppenthal discloses the method as recited in claim 5 wherein the first and second numbers are n-bit numbers, n being a positive integer and wherein the high order portion of the generated result is an n-bit portion. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 15, Huppenthal discloses the method as recited in claim 5 further comprising: in response to executing the subsequent single arithmetic instruction, multiplying third and fourth numbers specified by the subsequent single arithmetic instruction and adding implicitly the next partial result to generate a second result that represents the third number multiplied by the fourth number summed with the

next partial result. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 16, Huppenthal discloses the method as recited in claim 15, further comprising storing the high order portion of the second result to be implicitly added in response to executing another subsequent single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 18, Huppenthal discloses a method implemented in a device supporting a cryptography application, the method comprising: in response to executing a single arithmetic instruction of a processor instruction set, and storing at least a portion of the generated result; and using the stored at least a portion of the generated result in a subsequent computation in the cryptography application, and wherein the

single arithmetic instruction does not include an explicit source operand for specifying the partial result. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses multiplying a first number by a second number; adding implicitly a partial result from an executed single arithmetic instruction, wherein the partial result comprises a high order portion of a result of the previously executed single arithmetic instruction; adding a third number to generate a result that represents the first number multiplied by the second number summed with the partial result and the third number. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

Chen discloses supporting a cryptographic application as stated in Claim 1 above.

With Regards to Claim 19, Huppenthal discloses the method as recited in claim 18.

Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Hinds discloses performing the adding of the partial result as part of addition performed for the multiplying of the first and second number. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 21, Huppenthal discloses the method as recited in claim 18. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses performing the adding of the third number as part of the addition performed for the multiplying of the first and second number. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 22, Huppenthal discloses the method as recited in claim 18. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses adding the partial result comprises adding the partial result

after generation of a multiplication result of multiplying the first and second numbers. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 23, Huppenthal discloses the method as recited in claim 18. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses storing at least a portion of the generated result comprises storing a high order portion of the generated result as a next partial multiplication result for use with execution of a subsequent single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 30, Huppenthal discloses the method as recited in claim 24 wherein the second number is implicitly identified in the single arithmetic instruction.

(Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 33, Huppenthal discloses the method as recited in claim 18 wherein the first and third numbers are specified in the single arithmetic instruction as first and second source registers and a low order portion of the generated result is stored in a destination location specified in the single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 34, Huppenthal discloses the method as recited in claim 33 wherein the partial result from an executed single arithmetic instruction is implicitly specified by the single arithmetic instruction and wherein the second number is explicitly specified by a third source register in the single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements

can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 35, Huppenthal discloses the method as recited in claim 33 wherein the partial result from an executed single arithmetic instruction is implicitly specified by the single arithmetic instruction and the second number is implicitly specified by the single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 36, Huppenthal discloses the method as recited in claim 23 further comprising: in response to executing the subsequent single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)
Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Hinds discloses multiplying a fourth number and a fifth number, the fourth number being specified by the subsequent single arithmetic instruction, adding implicitly the next partial multiplication result and adding a sixth number to generate a second result, the second result representing the fourth number multiplied by the fifth number summed with the next partial result and the sixth number. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 37, Huppenthal discloses the method as recited in claim 36 wherein the fifth number and the second number are equal. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port; operands can be any values)

With Regards to Claim 38, Huppenthal discloses the method as recited in claim 36. Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Hinds discloses storing a high order portion of the second result to be implicitly added in response to executing another subsequent single arithmetic instruction. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 39, Huppenthal discloses the method as recited in claim 18. wherein the first number is specified in the single arithmetic instruction in a first source register, the second number is contained in a special register and is not specified in the single arithmetic instruction, the third number is specified as a second source register in the single arithmetic instruction and a low order portion of the generated result is stored in a destination location specified in the single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 40, Huppenthal discloses the method as recited in claim 18

wherein the first, second, and third numbers are specified by source operands in the single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port (source operands)) -

With Regards to Claim 41, Huppenthal discloses the method as recited in claim 18 wherein a destination location and one of the first number, the second number, and the third number are specified by one operand in the single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port (source operands))

With Regards to Claim 42, Huppenthal discloses the method as recited in claim 18. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses multiplying and adding operations are implemented for binary polynomial fields. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8,

lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 43, Huppenthal discloses a processor comprising an arithmetic circuit, the processor configured to be responsive to execution of a single arithmetic instruction of a processor instruction set, and wherein the single arithmetic instruction does not include an explicit source operand for specifying the partial result.
(Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)
Huppenthal does not specifically disclose a multiplying and summing sequence.
However, Hinds discloses to cause the arithmetic circuit to multiply a first number and a second number and to add implicitly a high order portion of a partial result from an executed single arithmetic instruction, thereby generating a result that represents the first number multiplied by the second number summed with the high order portion of the partial result; store at least a portion of the generated result; and use the stored at least a portion of the generated result in a subsequent computation. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and

adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose arithmetic operations is as stated in Claim 1 above.

With Regards to Claim 44, Huppenthal discloses the processor as recited in claim 43. Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses wherein to store at least a portion of the generated result, the processor is further responsive to the single arithmetic instruction to store a high order portion of the generated result into an extended carry register for use with execution of a subsequent single arithmetic instruction of a processor instruction set. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder))

Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

With Regards to Claim 49, Huppenthal discloses the processor as recited in claim 43 wherein the first and second numbers are specified in the single arithmetic instruction as first and second source registers. (Huppenthal col. 3, lines 1-7: number of MAP

elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 50, Huppenthal discloses a processor comprising an arithmetic circuit, the processor configured to be responsive to execution of a single arithmetic instruction of a processor instruction set, and wherein the single arithmetic instruction does not include an explicit source operand for specifying the partial result.

(Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Hinds discloses causing the arithmetic circuit to multiply a first number and a second number to add a third number and to implicitly add a high order portion of a result from an executed single arithmetic instruction thereby generating a result that represents the first number multiplied with the second number, summed with the high order portion of the previous result and with the third number; store at least a portion of the generated result; and use the stored at least a portion of the generated result in a

subsequent computation. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)) Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

With Regards to Claim 51, Huppenthal discloses the processor as recited in claim 50 wherein to store at least a portion of the generated result, the processor is configured to store a high order portion of the generated result for use with execution of a subsequent single arithmetic instruction of a processor instruction set. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 56, Huppenthal discloses the processor as recited in claim 50, wherein the first number is specified in the single arithmetic instruction as a first source register, the second number is contained in a logically local register and is not specified in the single arithmetic instruction, the third number is specified as a second source register in the single arithmetic instruction and a low order portion of the result is stored

in a destination location specified in the single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 57, Huppenthal discloses a computer-readable storage medium, comprising program instructions executable by a processor: wherein a single arithmetic instruction of a processor instruction set in the cryptography application, and wherein the single arithmetic instruction does not include an explicit source operand for specifying the high order portion of the result of the previously executed single arithmetic instruction. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port) Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses to cause the processor to multiply a first number by a second number and to implicitly add a high order portion of a result of a previously executed single arithmetic instruction to generate a result that represents the first number multiplied with the second number and summed with the high order portion of an

executed single arithmetic instruction, wherein the single arithmetic instruction further causes the processor to store a high order portion of the generated result for use with execution of a subsequent single arithmetic instruction in the cryptography application. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)
Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.
Chen discloses a cryptographic application as stated in Claim 1 above.

With Regards to Claim 58, Huppenthal discloses the storage medium as recited in claim 57 wherein the single arithmetic instruction includes a first source operand and a second source operand, specifying the first number and the second number, and a destination operand wherein execution of the single arithmetic instruction further causes the processor to store a low order portion of the generated result in a location specified by the destination operand. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

With Regards to Claim 59, Huppenthal discloses the storage medium as recited in claim 57.

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses execution of the subsequent single arithmetic instruction causes the processor executing the subsequent single arithmetic instruction to multiply a third number by a fourth number and implicitly add the high order portion of the result. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

With Regards to Claim 60, Huppenthal discloses the storage medium as recited in claim 59.

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses wherein another single arithmetic instruction of a processor instruction set in the cryptography application causes the processor to multiply a fifth number by a sixth number and to generate another result without implicitly adding another high order portion of another executed result and to store a high order portion of the other result for use with another subsequent single arithmetic instruction of a processor instruction set. (Hines col. 3, lines 5-17: applying a multiply-accumulate

operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder) Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above. Chen discloses the cryptographic application as stated in Claim 1 above.

With Regards to Claim 61, Huppenthal discloses a computer-readable storage medium, comprising program instructions executable by a processor to implement a cryptography application: wherein a single arithmetic instruction of a processor instruction set in the cryptography application, and wherein the single arithmetic instruction does not include an explicit source operand for specifying the partial multiplication result. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port) Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses to cause the processor to: multiply a first number by a second number; add implicitly a partial multiplication result from an executed single arithmetic instruction and a third number to generate a result that represents the first number multiplied by the second number summed with the partial multiplication result and

summed with the third number; and store a high order portion of the generated result for use with execution of a subsequent single arithmetic instruction of a processor instruction set in the cryptography application. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

Chen discloses the cryptographic application as stated in Claim 1 above.

With Regards to Claim 62, Huppenthal discloses the storage medium as recited in claim 61.

Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Hinds discloses execution of the subsequent second single arithmetic instruction causes the processor to multiply a fourth number by the second number to add a fifth number, and to implicitly add the high order portion of the generated result. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

With Regards to Claim 63, Huppenthal discloses the storage medium as recited in claim 61.

Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Hines discloses execution of the subsequent single arithmetic instruction causes the processor to multiply a fourth number by a fifth number, to add a sixth number, and to implicitly add the high order portion of the result. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

With Regards to Claim 64, Huppenthal discloses a processor supporting a cryptography application comprising: means, responsive to a single multiply-accumulate instruction of a processor instruction set in the cryptography application, and wherein the single arithmetic instruction does not include an explicit source operand for specifying the partial result. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via

chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses multiplying a first number with a second number and implicitly adding a partial result of an executed single multiply-accumulate instruction of a processor instruction set to generate a result that represents the first number multiplied by the second number summed with the partial result; and means for storing a high order portion of the result for use with execution of a subsequent single multiply-accumulate instruction of a processor instruction set in the cryptography application. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

With Regards to Claim 65, Huppenthal discloses a processor supporting a cryptography application, comprising: means, responsive to a single multiply-accumulate instruction of a processor instruction set in a cryptography application, and wherein the single arithmetic instruction does not include an explicit source operand for specifying the partial result. (Huppenthal col. 3, lines 1-7: number of MAP elements

chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port)

Huppenthal does not specifically disclose a multiplying and summing sequence. However, Hinds discloses multiplying a first number with a second number, for implicitly adding a partial result of an executed single multiply-accumulate instruction, and for adding a third number to generate a result that represents the first number multiplied by the second number summed with the partial result and the third number; and means for storing a high order portion of the generated result for use with execution of a subsequent multiply-accumulate instruction of a processor instruction set in the cryptography application. (Hines col. 3, lines 5-17: applying a multiply-accumulate operation to operands; multiplying operands and adding multiplication results; col. 8, lines col. 8, lines 6-25: chained multiply-accumulate operation; carry save adders and usage of partial multiplier (partial results: high order bits); output of results of partial multiplier is normalized and passed to carry save adders and final product adder)

Motivation for Hinds to disclose a chained operation is as stated in Claim 1 above.

6. Claims **3, 20, 45, 52** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal-Hinds-Chen** and further in view of **Lasher et al.** (US Patent No. **4,863,247**).

With Regards to Claim 3, Huppenthal discloses the method as recited in claim 1. Hinds discloses the partial result as stated in Claim 1 above.
Huppenthal-Hinds does not specifically disclose redundant number representation. However, Lasher discloses wherein the result is in redundant number representation. (see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)

It would have been obvious to one of ordinary skill in the art to modify Chen for a result in redundant number representation as taught by Lasher. One of ordinary skill in the art would have been motivated to employ the teachings of Lasher that fully parallel carry-free operation is provided for with reduced complexity. (see Lasher col. 2, lines 57-62)

With Regards to Claim 20, Huppenthal discloses the method as recited in claim 18, Hinds discloses the partial result as stated in Claim 1 above.
Huppenthal-Hinds does not specifically disclose redundant number representation. However, Lasher discloses wherein the result is in redundant number representation. (see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)
Motivation for Lasher to disclose redundant number representation is as stated in Claim 3 above.

With Regards to Claim 45, Huppenthal discloses the processor as recited in claim 43. However, Hinds discloses the previously executed single arithmetic instruction and the

high order portion of the executed single arithmetic instruction is stored as stated in Claim 1 above.

Huppenthal-Hinds does not specifically disclose redundant number representation.

However, Lasher discloses wherein the portion is stored in a redundant number representation. (see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)

Motivation for Lasher to disclose redundant number representation is as stated in Claim 3 above.

With Regards to Claim 52, Huppenthal discloses the processor as recited in claim 50.

Hinds discloses wherein the high order portion of the previous result as stated in Claim1 above.

Huppenthal does not specifically disclose a multiplying and summing sequence.

However, Lasher discloses wherein the portion is stored in redundant number representation. (see Lasher col. 6, lines 6-9; col. 6, lines 16-18: redundant number representations)

Motivation for Lasher to disclose redundant number representation is as stated in Claim 3 above.

7. Claims **6 - 12, 24 - 29, 31, 32, 48, 53, 54, 55** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal-Hinds-Chen** and further in view of **Stribaek et al.** (US Patent No. **7,181,484**).

With Regards to Claim 6, Huppenthal discloses the method as recited in claim 5. Hinds discloses wherein said storing the high order portion of the generated result comprises storing the high order portion of the generated result into a register for use with execution of the subsequent single arithmetic instruction as stated in Claim 1 above Huppenthal-Hines does not specifically disclose an extended carry register.
However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: load value extended carry operations)

It would have been obvious to one of ordinary skill in the art to modify Chen as taught by Stribaek for usage of an extended carry register. One of ordinary skill in the art would have been motivated to employ the teachings of Stribaek to enable the capability for extended precision in arithmetic calculations due to extensive and increasing usage of public key cryptography. (see Stribaek col. 1, lines 61-67)

With Regards to Claim 7, Huppenthal discloses the method as recited in claim 6. Hinds discloses retrieving an indication of a current value of the register by executing another single arithmetic instruction that multiplies a third number by a fourth number and that implicitly adds current contents of the extended carry register to generate a second result that represents the third number multiplied by the fourth number summed with the current contents of the register as stated in Claim 1 above.
Huppenthal-Hinds does not specifically disclose an extended carry register.
However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5,

lines 41-45: load value extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 8, Huppenthal discloses the method as recited in claim 7.
However, Hinds discloses a low order portion of the second result contains the indication of the current value of the register as stated in Claim 1 above.
Huppenthal-Hinds does not specifically disclose an extended carry register.
However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: load value extended carry operations)
Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 9, Huppenthal discloses the method as recited in claim 7, wherein the third and fourth numbers are zero. (Huppenthal col. 3, lines 1-7: number of MAP elements chained together to accomplish a single function or operation (implies a single arithmetic instruction; col. 3, lines 18-25: MAP elements can receive operands via chained port; col. 18, line 66 - col. 19, line 3: output data from one MAP element to be sent directly to the user array of the next MAP element with no processor intervention via a chain port; operands can be any values)

With Regards to Claim 10, Huppenthal discloses the method as recited in claim 6.
However, Hinds discloses loading the register with a predetermined value by executing another single arithmetic instruction that multiplies a third number by a fourth number

and that implicitly adds a current value of the extended carry register, to generate a result that represents the third number multiplied by the fourth number summed with the current value of the register, thereby loading the register with the predetermined value as stated in Claim1 above.

Huppenthal-Hinds does not specifically disclose an extended carry register. However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: load value extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 11, Stribaek discloses the method as recited in claim 6, further comprising selecting one of a plurality of extended carry registers as the extended carry register. (see Stribaek col. 5, lines 41-45: extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 12, Huppenthal discloses the method as recited in claim 6, However, Hinds discloses accessing the register via at least one of a load instruction and a store instruction as stated in Claim 1 above.

Huppenthal-Hinds does not specifically disclose an extended carry register.

However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 24, Huppenthal discloses the method as recited in claim 23. Hinds discloses storing the high order portion of the generated result comprises storing the high order portion of the generated result into a register for use with execution of the subsequent arithmetic instruction as stated in Claim1 above.

Huppenthal-Hinds does not specifically disclose an extended carry register.

However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 25, Huppenthal discloses the method as recited in claim 24. Hinds discloses retrieving an indication of a current value of the register by executing another single arithmetic instruction that multiplies a fourth number by a fifth number, that implicitly adds current contents of the register, and that adds a sixth number to generate a second result that represents the fourth number multiplied by the fifth number summed with the current contents of the register and the sixth number as stated in Claim 1 above.

Huppenthal-Hines does not specifically disclose an extended carry register.

However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 26, Huppenthal discloses the method as recited in claim 25,

Hinds discloses a low order portion of the second result contains the indication of the current value of the register as stated in Claim1 above.

Huppenthal-Hinds does not specifically disclose an extended carry register.

However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 27, Huppenthal discloses the method as recited in claim 24. Hinds discloses loading the register with a predetermined value by executing another single arithmetic instruction that multiplies a fourth number by a fifth number; and implicitly adds a current value of the register and adds a sixth number, to generate a result that represents the third number multiplied by the fourth number summed with the current value of the register and summed with the sixth number and to store it in the register, thereby loading the register with the predetermined value as stated in Claim 1 above.

Huppenthal-Hinds does not specifically disclose an extended carry register. However, Stribaek discloses wherein an extended carry register. (see Stribaek col. 5, lines 41-45: extended carry operations)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 28, Stribaek discloses the method as recited in claim 24, further comprising selecting one of a plurality of extended carry registers as the

extended carry register. (see Stribaek col. 5, lines 41-45: extended carry operations)
Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 29, Stribaek discloses the method as recited in claim 24, further comprising accessing the extended carry register via at least one of a load instruction and a store instruction. (see Stribaek col. 5, lines 41-45: extended carry operations)
Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 31, Huppenthal discloses the method as recited in claim 30.
Hinds discloses accessing a register storing the second number via at least one of a load instruction and a store instruction as stated in Claim 1 above.
Stribaek discloses wherein a special register as stated in Claim 6 above.

With Regards to Claim 32, Huppenthal discloses the method as recited in claim 18.
Hinds discloses accessing a special register storing the second number via at least one of a load instruction and a store instruction as stated in Claim 1 above.
Stribaek discloses a special register as stated in Claim 6 above.

With Regards to Claim 48, Stribaek discloses the processor as recited in claim 43, wherein the extended carry register is a special register. (see Stribaek col. 5, lines 41-45: extended carry register)

Motivation for Stribaek to disclose extended carry register is as stated in Claim 6 above.

With Regards to Claim 53, Huppenthal discloses the processor as recited in claim 50. Hinds discloses the processor is configured to store the high order portion of the generated result into a register as stated in Claim 1 above.

Stribaek discloses wherein an extended carry register as stated in Claim 6 above.

With Regards to Claim 54, Huppenthal discloses the processor as recited in claim 50. Hinds discloses the register is a special register accessible by the processor via at least one of load instruction and store instructions as stated in Claim 1 above.

Stribaek discloses extended carry register as stated in Claim 6 above.

With Regards to Claim 55, Huppenthal discloses the processor as recited in claim 50. Hinds discloses wherein the register has an associated dirty bit indicating whether contents of the register need to be saved on a context switch as stated in Claim 1 above.

Stribaek discloses an extended carry register as stated Claim 6 above.

8. Claim **17** is rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal-Hinds-Chen** and further in view of **Chen et al.** (US Patent No. **6,687,725**: referred to as "Chen2").

With Regards to Claim 17, Huppenthal discloses the method as recited in claim 1. Hinds discloses multiplying and adding are implemented in Claim 1 above.

Huppenthal-Hinds does not specifically discloses XOR operations.

However, Chen2 discloses wherein to support XOR operations for binary polynomial fields. (see Chen2 col. 4, line 64 - col. 5, line 2; col. 15, lines 29-31: XOR operations)

It would have been obvious to one of ordinary skill in the art to modify Chen to support XOR operations as taught by Chen2. One of ordinary skill in the art would have been motivated to employ the teachings of Chen2 to provide an arithmetic circuit which can perform all arithmetic operations in the finite field, including addition, multiplication, division, exponentiation and inverse multiplication. (see Chen2 col. 3, lines 17-21)

9. Claims **46, 47** are rejected under 35 U.S.C. 103(a) as being unpatentable over **Huppenthal-Hinds-Chen-Lasher** and further in view of **Stribaek**.

With Regards to Claim 46, Huppenthal discloses the processor as recited in claim 45. Stribaek discloses an extended carry register as stated in Claim 6 above.

With Regards to Claim 47, Huppenthal discloses the processor as recited in claim 45. Stribaek discloses wherein an extended carry register as stated in Claim 6 above.

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Carlton V. Johnson whose telephone number is 571-270-1032. The examiner can normally be reached on Monday thru Friday , 8:00 - 5:00PM EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Nasser Moazzami can be reached on 571-272-4195. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

/David García Cervetti/
Primary Examiner, Art Unit 2436

Carlton V. Johnson
Examiner
Art Unit 2436

CVJ
July 6, 2010